

REMARKS

Entry of this response is proper under 37 CFR §1.116, since no claim amendments or new issues are raised herein.

Claims 1, 2, and 4-27 are all the claims presently pending in the application. Claim 3 is canceled.

It is noted that composite blocking is the more generic concept of double blocking addressed in claims 4, 11, 17, 20, 24, and 27, which claims are not rejected under the prior art evaluation currently of record. Applicants, therefore, conclude that the Examiner considers these claims to be allowable pending resolution of the issue of non-statutory subject matter.

It is noted that Applicants specifically state that no amendment to any claim herein should be construed as a disclaimer of any interest in or right to an equivalent of any element or feature of the amended claim.

Claims 1, 2, and 4-27 stand rejected under 35 U.S.C. § 101 as allegedly directed to non-statutory subject matter. Claims 1, 2, 5-10, 12-16, 18, 19, 21-23, and 25-27 stand rejected under 35 U.S.C. § 112, second paragraph, as allegedly indefinite because the Examiner does not understand the meaning of “storing data contiguously.”

Claims 1, 2, 5-10, 12-16, 18, 19, 21-23, and 25-27 stand rejected under 35 U.S.C. § 103(a) as unpatentable over U.S. Patent No. 5,099,447 to Myszewski, further in view of US Patent Publication 2003/0088600 to Lao et al.

These rejections are respectfully traversed in the following discussion.

I. THE CLAIMED INVENTION

The claimed invention is directed to method of increasing at least one of efficiency and speed in executing a matrix subroutine on a computer. Data for a matrix subroutine call is stored as contiguous data in a computer memory in an increment block size that is based on a cache size of the computer. A first dimension of the block is larger than a corresponding first dimension of the cache and a second dimension of the block is smaller than a corresponding second dimension of the cache, so that the block fits into a working space of the cache.

Since standard Fortran and C two-dimensional arrays are stored in memory/processed in either a column or a row of the matrix data, depending on which language is used, there are inefficiencies in retrieving data for sub-blocks of matrix data, as discussed in more detail below.

In contrast, the present invention teaches that the data is to be stored in memory as a collection of sub-blocks of data that each fits into the cache working area and that is stored contiguously in memory, to preclude thrashing of data in cache.

By actually storing in memory the matrix data as contiguous submatrix data and by sizing the submatrix block of data so that one dimension of the submatrix block exceeds the dimension of the cache relative to the size of the cache (a number of submatrix blocks make up a square matrix), the processing is executed faster than conventional methods, since data thrashing is reduced or eliminated.

Another aspect of the present invention is the recognition that many engineering/scientific matrices are typically directed toward square matrices, since such square matrices express linear equations in which a specific solution can be found. In contrast to conventional wisdom, the present invention also teaches that such square matrices can be viewed as rectangular blocks of matrix data that can be brought into a working area of cache as a block of contiguous data, again, thereby reducing or eliminating the thrashing that typically occurs when the entire square block is retrieved. The composite block of the present invention allows matrix data that, in modern engineering/scientific applications, typically involve a matrix having at least one dimension that exceeds either dimension of the working area of the cache, to be brought into cache in contiguous blocks that address the problem that plagues the art of cache data thrashing.

As an example of increased processing efficiency, an application related to the square blocking of the present invention using contiguous DGEMM operands is Cholesky factorization. Cholesky factorization is an example of a DLAF. On the IBM Power3 processor the implementation of Cholesky factorization using the present invention achieves 92% of peak performance whereas conventional full format LAPACK DPOTRF achieves 77% of peak performance. Moreover, all programming for the new data structures discussed in the present invention can be accomplished in standard Fortran (or C/C++), through the use of higher dimensional full format arrays. Thus, no new compiler support is necessary to implement the present invention.

II. THE 35 USC §101 REJECTION

The Examiner continues to reject claims 1, 2, and 4-27 under 35 U.S.C. §101 as allegedly directed toward non-statutory subject matter as allegedly failing to provide a real world result.

Again, Applicants respectfully point out that claim 1 describes a method that increases the efficiency and/or running time of a matrix subroutine which is useful and practical. It does so, by changing how the matrix is stored in the memory of the computer.

On the bottom line on page 2 of the Office Action, the Examiner alleges: “*Neither the data stored in the caches nor the result of executing matrix subroutine has a real world value.*”

Applicants respectfully submit that this is a false statement. How the matrix data gets into and out of the cache greatly effects how efficient the matrix subroutine will execute.

It is implicit in claim 1 that the matrix is stored in one of the two standard formats of DLA. In the paper “High-performance linear algebra algorithms using new generalized data structures for matrices” by Fred Gustavson, it is shown that these standard data structures hurt the performance of matrix subroutines. These results are generally accepted by the Engineering and Scientific Community at large.

Broadly speaking, Claim 1 describes a better data structure for a matrix that the matrix subroutine will be processing. The result will be a more efficient execution of the matrix subroutine. Claim 1 is directed at changing the standard input data to the better matrix data structure of the invention.

The Examiner also seems to suggest that, in order for a claim involving matrix manipulation to pass muster as statutory subject matter, the claim must recite a real-world application being processed using that matrix processing. If this implied requirement is what the Examiner is actually attempting to say in this rejection, then Applicants bring the Examiner’s attention to the USPTO Memorandum from Deputy Commissioner John Love, dated April 12, 2007, and addressed to Technology Center Directors and having the subject: “Clarification of Interim Guidelines For Examination of Patent Applications for Subject Matter Eligibility.”

In that memorandum, Deputy Commissioner Love states: “*It is the result that should be the focus. If the result has a real world practical application/use, then the test has been satisfied. The claim need not include the uses to which the result is ultimately put, just the result itself.*”

Therefore, taking claim 1 as an example, the result of the claimed invention could be

viewed either as the “increasing of efficiency/speed” or as the defined step for store data, as recited in the claim limitation. According to Deputy Commissioner Love, there would be no need to recite a use of the processing being achieved by the matrix multiplication, since the result is in the increase in speed and/or efficiency of the processing.

It is brought to the Examiner’s attention that this increase in speed/efficiency can be physically measured and so, inherently has “real-world” application.

Therefore, again, Applicants respectfully submit that the claimed invention meets the criterion for statutory subject matter because its result is that of increasing computer efficiency, using the method described in the independent claims.

In view of the foregoing, the Examiner is respectfully requested to reconsider and withdraw these rejections for non-statutory subject matter.

III. THE REJECTION UNDER 37 CFR §112, SECOND PARAGRAPH

The Examiner continues to maintain the rejection for claims 1, 2, 5-10, 12-16, 18, 19, 21-23, and 25-27 under 37 CFR §112, second paragraph, because the Examiner considers that the claim language “... storing data contiguously ... in an increment block size ...” is somehow unclear, since, the Examiner continues “... data of a matrix contiguously stored in memory can be view as being stored in any increment size.”

In response, Applicants respectfully point out that claim 1 states that the data structure of the present invention is contiguous in memory. Standard data structures for matrices store sub matrices memory as non-contiguous data.

The concept of contiguous data storage as related to the blocks of the present invention is actually quite well described at various locations in the specification. For example, lines 6-8 of page 12 recite: “*Each of the four $NB/2$ by $2NB$ sub-rectangles 301, which are stored row-wise, is preferably contiguous in memory and map well into the L1 cache.*”

Further, lines 14-18 on page 14 recite: “*Although it might seem intuitive that a block size based on double the dimension of the L1 cache would hinder performance for matrix multiplication, the reason that this approach works is that the portion of the matrix brought into the cache with each block allows a complete multiplication of that block.*”

Lines 7-12 of page 15 recite: “*Typically, in a machine in which memory is accessed in*

increments of line size, preferably the matrix data is laid out contiguously in memory in “stride one” form, where “stride one” means that the data is retrieved in increments of a line.

Therefore, in machines having a line size, according to the present invention, the matrix data is preferably contiguously stored into and preferably retrieved from memory in increments of the machine line size.”

Thus, the present invention involve data contiguously stored in memory so that an entire block of size $NB/2$ by $2NB$ will be substantially retrieved in a continuous and uninterrupted memory operation and some computers will actually transfer this block size of data as a single memory operation.

Therefore, Applicants respectfully submit that one having ordinary skill in the art would clearly understand the meaning of this claim terminology, particularly in view of the description in the specification.

Thus, in contrast to matrix data as typically stored in memory, the present invention teaches to bring an entire block of data into cache, where the block size has been predetermined to fit into the cache working area even though the block has one dimension larger than the cache working area. There is no suggestion in the references of record to place matrix data in contiguous order in blocks of data to be moved into cache in units of these blocks having the dimensions described in the independent claims.

Therefore, Applicants respectfully submit that there is no lack of clarity in the claim wording when viewed in its entirety and respectfully request that the Examiner reconsider and withdraw this rejection.

IV. THE PRIOR ART REJECTION

The Examiner continues to maintain that Myszewski, when modified by Lao, renders obvious the claimed invention described by claims 1, 2, 5-10, 12-16, 18, 19, 21-23, and 25-27.

Applicants again respectfully disagree and submit that the rejection of record fails to meet the initial burden of a *prima facie* obviousness rejection, since, even if the two cited references were to be combined, the combination would still fail to satisfy the plain meaning of the claim language of even the independent claims.

That is, the present invention describes a general technique of dividing data into large

data blocks related to the size of the working area of the cache. Moreover, as clearly described in the independent claims, the blocks of data of the present invention involve blocks having one dimension larger than the cache dimension and one dimension smaller than the cache dimension.

Neither reference cited by the Examiner suggests a block size with such dimensions and the Examiner makes no attempt to point to any description to support the rejection. The description in paragraph [0071] of secondary reference Lao and the example therein does not satisfy the description of the block size described in the independent claims, since the partitions shown are related to the matrix size, not the cache size. Nor is there is no suggestion in paragraph [0071] to have a block dimension larger than the cache dimension.

Hence, turning to the clear language of the claims in neither Myszewski nor Lao is there any teaching or suggestion of: "...a first dimension of said block being larger than a corresponding first dimension of said cache and a second dimension of said block being smaller than a corresponding second dimension of said cache, such that said block fits into a working space of said cache", as required by independent claim 1. The remaining independent claims have similar language.

Moreover, Applicants respectfully submit that the Examiner does not understand the present invention. Both Myszewski and Lao are directed to matrices subroutines whose input matrices are in standard format. As Applicants have attempted to explain to the Examiner, it is impossible to partition a matrix in standard format in contiguous blocks unless one changes the layout of the standard format.

Neither Myszewski nor Lao does this in their patents.

Regarding LAPACK matrix routines, the present invention is claiming a way to make the LAPACK matrix routine more efficient as LAPACK requires its input matrices to be in standard format. Current LAPACK are improved by Myszewski. The present invention improves LAPACK routines even further in a non obvious manner.

Regarding Claim 23, the present invention is about sub-matrices that are contiguous. The current art of matrix subroutines expects its matrices to be in standard format.

Therefore, Applicants submit that there are elements of the claimed invention that are neither taught nor suggested in the prior art of record, and the Examiner is respectfully requested to reconsider and withdraw this rejection.

Additionally, Applicants submit again the following comments relative to this prior art

rejection, particularly since the prior art rejection does not address all claims.

The present invention teaches the technique of composite blocking, as articulated in the independent claims. The “double blocking” defined in claims 4, 11, 17, and 20 is a specific embodiment of composite blocking. There is no suggestion in Myszewski to convert and store in memory the matrix data of a matrix in square blocks larger than cache and consisting of rectangular blocks of contiguous data that will each fit into the working area of a cache (even though one dimension of the rectangular blocks is larger than its corresponding cache dimension).

To consider this prior art reference in a different perspective, a rectangular block makes DGEMM run faster. However, the DLAFA requires square blocking. The composite blocking of the present invention serves both purposes. Myszewski does not even recognize this disparate requirement and does not suggest a technique that satisfies this dual requirement, nor does it address DLAFA.

The technique of the present invention reduces the data thrashing that typically occurs when matrix data is simply retrieved in its original format as is required by higher level languages such as Fortran and C.

The data thrashing occurs, in the conventional processing, because portions of the matrix data are typically flushed from cache during processing so that additional memory accesses are required as the flushed data becomes needed for current processing and because matrix data is typically not contiguous as actually stored in memory. The present invention addresses this data thrashing by actually storing in memory the matrix data as rectangular contiguous sub blocks of a size that fits into cache and, if the matrix data is also contiguous, it will also be organized on memory line size, thereby increasing the speed at which future matrix data will enter the cache to be consumed in the matrix processing. This occurs because the data actually brought into cache from memory will be completely contiguous matrix data (rather than standard layout data in portions of some lines of data) and because the matrix subroutine processing will fit the processing result back into the data currently being consumed, rather than inadvertently flushing out matrix data not yet being processed.

In contrast to the conventional method of dealing with matrix data, the present invention teaches storing the matrix data in memory as actually being rectangular sub blocks of data that fits the sub blocks together so that all the matrix data is contiguous, where “contiguous” means that the data is in the order to be used in its processing. The standard format of higher level

conventional computer languages makes no attempt to place matrix data into its appropriate “contiguous” format.

Finally, relative to the Examiner’s Response in paragraph 6 beginning at the top of page 5 of the Office Action, Applicants respectfully submit that, regarding subparagraph 2, the Examiner’s statement is true. However, it is irrelevant, as the sub matrices of standard format are not contiguous in memory.

Regarding subparagraph 3, Myszewski must, in DGEMM, process standard matrix format as the API for DGEMM requires this format. In contrast, the current claim 1 is directed at a non standard format whose use will improve the efficiency of a matrix subroutine like DGEMM even further. Based on this observation, the Examiner’s remark appears true, but it is irrelevant relative to the present invention. A careful reading of both Myszewski and Lao will reveal that they only consider standard matrix data layout whose sub matrices are not contiguous.

Therefore, Applicants submit that there are elements of the claimed invention that are not taught or suggested by Myszewski, even if modified by Lao, and the Examiner is respectfully requested to withdraw this rejection.

V. FORMAL MATTERS AND CONCLUSION

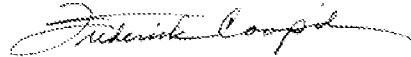
In view of the foregoing, Applicant submits that claims 1, 2, and 4-27, all the claims presently pending in the application, are patentably distinct over the prior art of record and are in condition for allowance. The Examiner is respectfully requested to pass the above application to issue at the earliest possible time.

Should the Examiner find the application to be other than in condition for allowance, the Examiner is requested to contact the undersigned at the local telephone number listed below to discuss any other changes deemed necessary in a telephonic or personal interview.

Serial No. 10/671,887
Docket No. YOR920030010US1 (YOR.424)

The Commissioner is hereby authorized to charge any deficiency in fees or to credit any overpayment in fees to Assignee's Deposit Account No. 50-0510.

Respectfully Submitted,



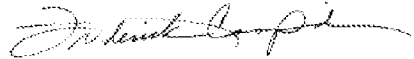
Date: December 31, 2007

Frederick E. Cooperrider
Registration No. 36,769

McGinn Intellectual Property Law Group, PLLC
8321 Old Courthouse Road, Suite 200
Vienna, VA 22182-3817
(703) 761-4100
Customer No. 21254

CERTIFICATION OF TRANSMISSION

I certify that I transmitted electronically, via EFS, this Amendment under 37 CFR §1.116 to Examiner C. Ngo on December 31, 2007.



Frederick E. Cooperrider
Registration No. 36,769